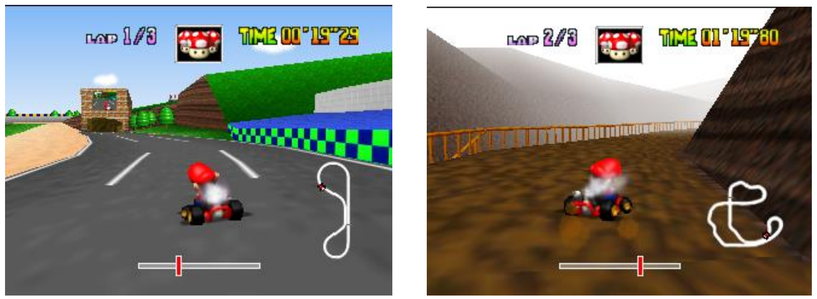
NeuralKart: A Real-Time Mario Kart 64 AI

Varun Ramesh (vramesh2), Eduardo Torres Montaño (torresm9), and Harrison Ho (h2o)

Introduction

- Many research efforts exist in developing autonomous vehicles.
- Mario Kart 64 represents a simplification of real-life autonomous driving, yet introduces interesting challenges, such as hazards, jumps, and items.



Examples of our bot racing in different levels. The bottom bar shows the current steering angle.

Previous Mario Kart 64 autopilot attempts:

- NEAT (Neural Evolution of Augmenting Topologies), a genetic algorithm, has been applied to Mario Kart with good performance. However, it "cheats" by reading game registers during real-time play, info not directly accessible to human players.
- TensorKart learns to map screenshots to human inputs. However, it requires unnatural human play and cannot recover from error conditions.

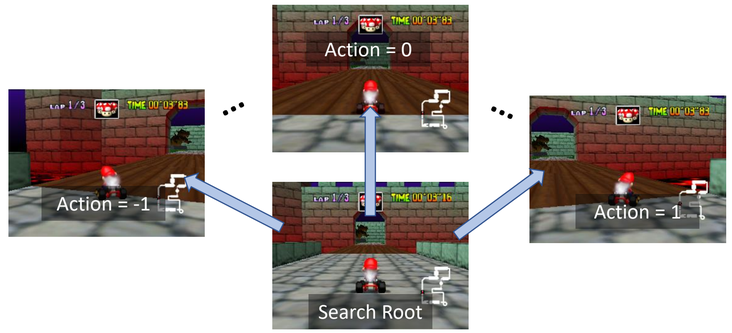
Problem Statement

- We developed a CNN model which plays Mario Kart 64 races in real time, taking a screenshot of the game and outputting a control stick angle.
- Our approach has two components:
  - A search-based AI with knowledge of game registers generates a dataset.
  - A CNN-based AI trains on the resulting dataset for real-time predictions.
  - We alternate between running the Search AI and CNN AI to explore different states.
- We use the BizHawk emulator to interface with the game during example generation/prediction.

Method/Model

Search based AI

- Every 30 frames, the AI checks 11 possible steering angles and chooses the angle yielding the greatest reward function.
- Simultaneously, the AI saves a picture associated with the steering angle, generating the dataset.
- Our dataset consists of 8,766 image-steering angle pairs and uses a 10% randomly selected validation split.



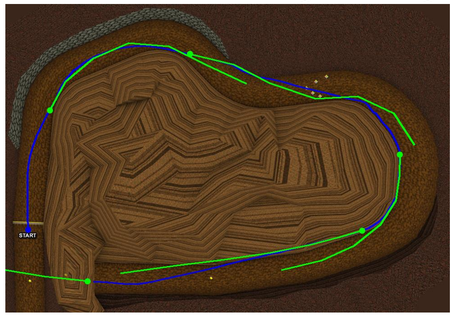
Demonstration of the search based AI. The AI will simulate each case and choose the steering angle maximizing the reward function.

CNN-based AI

- The CNN model is trained on the resulting dataset, to imitate the performance of the search AI.
- The model consists of 5 Batch Norm - 2D Convolution - ReLU layers, followed by 5 dense layers.

DAGGER (Dataset Aggregation) Algorithm

- The search AI rarely enters error states where it gets stuck, giving fewer examples to recover from/avoid.
- To account for this, we sample states from the CNN-based AI and run the search AI from those states, thus generating more training data.
  - CNN-based AI adds randomness and introduces new states to the autopilot.
  - Search AI uses omniscience to correctly respond to such states.

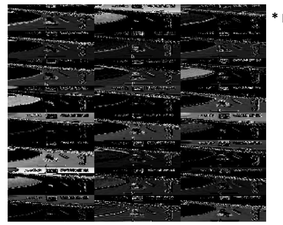


Example progress: green lines indicate search AI paths, and the blue line indicates the CNN path.

Findings

- We compared the AI's performance (10 run average) with that of a human over four different racetracks.
- The autopilot performs slightly worse than human players, but still yields competitive performance.

	Autopilot Time (s)	Human Time (s)
Moo Moo Farm	97.46	94.07
Luigi's Raceway	129.09, 1 Run DNF*	125.30
Choco Mountain	138.37, 2 Run DNF*	129.50
Rainbow Road	389.18	365.60



* Did not finish

The network detects features such as sand and walls. A game image is on the left, and the corresponding first layer activation functions viewed by the network are on the right.

- Shortcomings:
  - The search AI follows waypoints predefined in the game, which do not necessarily reflect the optimal racing strategy. As a result, our model will sometimes make wide, unnecessary turns.
  - The CNN does not understand risky situations and often drives close to hazards such as walls and grass, increasing error state probability.

Conclusion

- Our model shows that end to end neural systems can yield good performance in games such as Mario Kart 64.
- Possible future improvements we may look into:
  - Multiple time steps during training may give the CNN a sense of time and speed, improving racing decisions.
  - Reinforcement learning can reward good performance and punish error conditions, but would require additional coordination between the CNN and the game emulator, increasing software complexity.